テーマ１：指数関数時間アルゴリズム（その３）

本日の講義の目的

- 点彩色問題の指数時間アルゴリズム（その２＆その３）
- 包除原理

1.4. 包除原理に基づく点彩色問題の指数時間アルゴリズム

前回は点彩色問題に対して $O^*((2.4423\cdots)^n)$ 時間のアルゴリズムを示した．今回はそれをさらに改善するものを説明しよう．ここでは，次のような判定版の点彩色問題を考える．

点彩色問題（判定版） (COLOR: vertex COLORing problem)
入力： 無向グラフ $G=(V,E)$ と色数 K.
質問： G に対して K 色以下の頂点彩色が存在するか？

ここではこの問題を以下のような集合被覆問題（Set Cover Problem）と解釈して解法を考える．

集合被覆問題（直感的な説明用）
入力： 集合 V，集合族 $\mathcal{I}\subseteq\{I : I\subseteq V\}$，自然数数 K.
質問： Is there any set family $I_1,...,I_K$ of sets in $\mathcal{I}$ such that $V=I_1\cup\cdots\cup I_K$ holds?
補足：
(1) 上記の条件を満たす集合族を V の K-被覆族 と呼ぶことにする．
(2) 実際には $\mathcal{I}$ は大きすぎるので入力として与えることはできない．その意味でここに示したのは通常の「計算問題」とは少し異なる．

これからは，任意に選んだ無向グラフ $G=(V,E)$ と色数 K に対して話しを進める．$V=\{1,...,n\}$ とし，この n をサイズパラメータとして計算量を評価する．また，このグラフ G の独立点集合（互いに辺の無い集合）の族を $\mathcal{I}$ とする．この $\mathcal{I}$ を用いて，V の K-被覆集合族が構成できるかを議論しよう．そこで $\mathcal{I}$ の要素から構成される V の K-被覆集合族の個数を c_K とする．そうすると

G が K-頂点彩色可能 $\iff$ $c_K>0$（つまり，V の K-被覆集合族が 1 個以上ある）

となるのは明らかだろう．そこで，G の K-頂点彩色可能性を議論するために，c_K を求め，$c_K>0$ かを判定すればよい[1]．そこで以下では c_K を計算する方法を考え，それを $O^*(2^n)$ 時間で計算するアルゴリズムを示す．このことから直ちに次の定理が導かれる．

[1]ただし，前回のアルゴリズムと違い，$c_K>0$ が判定できたとして，それから点彩色問題（判定版）は計算できるが，それをどう使えば実際に K-頂点彩色を得られるかは，少し考えないとならない $\longrightarrow$ レポート課題の一部とする．

定理 1.6. 点彩色問題（判定版）は $O^*(2^n)$ で計算可能.

さて，c_K を求める方法だが，それに包除原理を用いる．これは（何らかの形で与えられる）集合族 $A_1, ..., A_m$ に対し，$A_1 \cup \cdots \cup A_m$ を計算する方法である．

我々の目標は $\mathcal{I}^K$ （つまり，$\mathcal{I}$ の要素の K 個組）の $(I_1, ..., I_K)$ 中で，$V = \cup_{k=1}^K I_k$ となるものの個数を数えることだ[2]．そこで，A_i を $\mathcal{I}^K$ の中で要素 $i \in V$ を含まない組の集合とする．つまり

$$A_i \ = \ \{\, (I_1, ..., I_K) : i \notin \cup_{k=1}^K I_k \,\}$$

とし，$N_i \stackrel{\text{def}}{=} \|A_i\|$ とする．同様に各 A_{i_1}, A_{i_2} に対し，$N_{i_1,i_2} \stackrel{\text{def}}{=} \|A_{i_1} \cap A_{i_2}\|$ と表わすことにしよう．また，$N \stackrel{\text{def}}{=} \|\mathcal{I}^K\|$ とする．すると

$$c_K \ = \ N - \|A_1 \cup A_2 \cup \cdots \cup A_n\|$$

となる．この $\|A_1 \cup A_2 \cup \cdots \cup A_n\|$ は，包徐原理で次のように計算することができる．

$$\|A_1 \cup A_2 \cup \cdots \cup A_n\| \ = \ \sum_{i \in [n]} N_i - \sum_{i_1 < i_2 \in [n]} N_{i_1,i_2} + \sum_{i_1 < i_2 < i_3 \in [n]} N_{i_1,i_2,i_3} - \cdots$$

以上をまとめると次のような計算式が得られる．ただし，以下では $N_W \stackrel{\text{def}}{=} \| \cap_{i \in W} A_i \|$ とし，$N_\emptyset \stackrel{\text{def}}{=} N$ とする．

$$c_K \ = \ \sum_{W \subseteq V} (-1)^{|W|} N_W. \tag{1}$$

そこで問題は N_W の計算となる．定義に戻ってみると，N_W は $(I_1, ..., I_K) \in \mathcal{I}^k$ の中で W の要素を含まない組の総数である．この条件を書き換えると，$(I_1, ..., I_K) \in \mathcal{I}^k$ が対象となるのは,

$$W \cap \cup_{k=1}^K I_k = \emptyset \iff \cup_{k=1}^K I_k \subseteq V - W \iff I_k \subseteq V - W \text{ for all } k = 1, ..., K$$

となる場合である．したがって，$\mathcal{I}(W) \stackrel{\text{def}}{=} \{I \in \mathcal{I} : I \subseteq V - W\}$ とすると，対象となる組の全体は $\mathcal{I}(W)^k$ の要素に他ならない．つまり

$$N_W \ = \ \|\mathcal{I}(W)\|^k.$$

となる．したがって，各 $W \subseteq V$ に対して $\|\mathcal{I}(W)\|$ を求める問題に帰着された.

ここでは, 動的計画法を用いて $\|\mathcal{I}(W)\|$ を求めてみよう．そのためには，うまく $\|\mathcal{I}(W)\|$ を求める再帰的定義が欲しい．それには関数の決め方が鍵となる．天下り的ではあるが, 次のような関数を考えよう.

$$f(i, W) \ \stackrel{\text{def}}{=} \ \|\{\, I \in \mathcal{I} \,:\, \{i+1, ..., n\} - W \subseteq I \subseteq V - W \,\}\|.$$

[2]ここでは，たとえば $I_1 \neq I_2$ のとき，(I_1, I_2, I_3) と (I_2, I_1, I_3) は異なるものとして数える．本当の c_K を計算するときは，これらは 1 個と数えなければならないが，ここでは $c_K > 0$ かどうかが問題なので，とくに気にしないことにする.

これに対して次の性質は明らか，と言ってもいいだろう．

(1) $\|\mathcal{I}(W)\| = f(n, W)$,
(2) $f(0, W) = 1$ (if $V - W \in \mathcal{I}$) and $f(0, W) = 0$ (otherwise),
(3) $f(i, V) = 0$.

とくに (2), (3) が帰納法の初期段階にあたる．一方，帰納段階にあたる計算は次のようになる．

$$f(i, W) \;=\; \begin{cases} f(i-1, W), & \text{if } i \in W, \text{ and} \\ f(i-1, W) + f(i-1, W \cup \{i\}), & \text{otherwise.} \end{cases}$$

これを表を使って再計算を防ぎながら計算する．そうすれば，表の要素数は $n \times 2^n$ なので以下の上界が得られる．

補題 1.7. $\|\mathcal{I}(W)\|$ は $O^*(2^n)$ 時間で計算可能.（ただし，上の方法では領域も $O^*(2^n)$ 必要.）

この補題ならびに (1) を使えば，定理 1.6 が導かれる．

1.5. 多項式領域アルゴリズム

これまで見てきた指数時間アルゴリズムは，計算時間だけでなく必要な計算領域（メモリ）量も指数関数的に増加してしまうアルゴリズムだった．それでは，すぐに計算領域を食いつぶして動かなくなってしまう．それに対し，たとえ指数時間かかっても，計算領域（メモリ）が妥当な範囲に収まる場合には何とか計算を続けられる場合がある．そこで，作業メモリ領域量（以下，領域量）が，サイズパラメータの多項式以下で収まる指数時間アルゴリズムの設計を考えてみよう．

ここでも点彩色問題（判定版）を用いる．アルゴリズムは本質的に上記の包除原理を使ったアルゴリズムを基本とする．すなわち，式 (1) を用いて c_K を計算するところまでは同じだ．そうすると重要なのは $\|I(W)\|$ の計算である．これは要するに，与えられたグラフ $G' = (V - W, E \cap ((V - W) \times (V - W)))$ における独立点集合の数を勘定する計算だ．これについては次の事実が知られている（参考文献 [1]）．

補題 1.8. 頂点数 n' のグラフの独立点集合の数は，多項式領域量でかつ $O^*(1.246^{n'})$ 時間で計算可能.

これを用いて点彩色問題を解く多項式領域量のアルゴリズムを作ることができる．

定理 1.9. 点彩色問題（判定版）は多項式領域量かつ $O^*(2.246^n)$ 時間で計算可能.

証明： 式 (1) を用いる際に $\|\mathcal{I}(W)\|$ の計算が必要になり，その都度，上の補題のアルゴリズムを用いれば $O^*(1.246^{n-\|W\|})$ 時間で計算ができる．一方，各 $i \in [W]$ に対し，$\|W\| = i$ となる W は $\binom{n}{n-i} = \binom{n}{i}$ 通り．したがって計算時間の総計は

$$\sum_{i=0}^{n} \binom{n}{n-i} 1.246^{n-i} \;=\; \sum_{i=0}^{n} \binom{n}{i} 1.246^{i} \;=\; (1 + 1.246)^n.$$

定理はこの解析から直ちに導かれる． □

参考文献

補題 1.8 の証明のアイデアは以下の論文から（学内ならば東工大の電子図書館を通して得ることができる）.

[1] M. Fürer ad S.P. Kasiviswanathan, Algorithms for counting 2-SAT solutions and coloring with applications, in *Proc. 3rd Int'l Conf. on Algorithmic Aspects in Information and Management* (AAIM'2007), Lecture Notes in Computer Sci, Springer, 4508, Springer, pp. 47–57.

テーマ＃１での課題（〆切：原則として６月１５日（金），ただし学期末でもＯＫ）

次のいずれかの問いに対する答えをレポートしてまとめる．参考文献，参考資料などを用いた場合は明記すること.

1. 与えられたグラフと K に対し，K 色を使った点彩色を１つ求める（もしあれば）アルゴリズムで，領域計算量が頂点数 n の多項式，時間計算量が適当な定数 $c_1, 0 < c_1 < 3$, のもとで，頂点数 n に対して $O^*(c_1^n)$ となるようなものを設計したい．その際，今回の講義で省略した次の点について明確に説明せよ.
   (1) 与えられたグラフと K に対して，c_K を計算することができたとして，その計算法を用いて，点彩色を実際に求めるアルゴリズムを示せ.
   (2) 与えらた n 頂点のグラフに対して，その独立点集合の数を求める多項式領域量かつ $O^*(c_2^n)$ 時間計算量のアルゴリズムを示せ．ただし，c_2 は $0 < c_2 < 2$ を満たす適当な定数であればよい.（注：要するに補題 1.8 を示せばよい.）
2. 与えられた無向グラフ $G = (V, E)$ と頂点 $s, t \in V$ に対し，s から t へのハミルトン路を求めるアルゴリズムで，多項式領域量かつ $O^*(c^n)$ 時間計算量（ただし $c > 0$ は適当な定数）を持つものを示せ.